ニューラルネット

-GA for Knapsack-

055702B 池野谷克俊

2007年 7月 25日 水曜日

1　課題内容

GA でナップサック問題を解く。その際、

(1) 各遺伝子操作オペレータ (選択、交叉、突然変異) を実装した場合

(2) 実装しない場合

について実験を行い、得られた結果の違いから各オペレータの役割を考察せよ。

2　解答及び考察

プログラムを実行する前に、制約条件 (KP.h)、荷重リスト (KP_List_original.data) を変更する。以下にその内容を示す。

制約条件 (KP.h)

```
//for KP_List_original.data
#define BAG_SIZE 300
#define ITEM_NUM 12
```

荷重リスト (KP_List_original.data)

```
10 8
20 20
30 22
40 28
50 40
65 36
25 40
45 50
70 34
30 15
75 50
65 43
```

さらに、終了世代数を 20 として実行する。

2.1　全て実装した場合

以下に全て実装した場合の実行結果を示す。

実行結果

```
[j05002@ga-knapsack]% ./run_ga.sh 1 1 KP_List_original.data 20
# reading args => seed_pop=1, seed_ga=1, itemfile=KP_List_original.data,
max_generation=20
gene no.0 = 0 1 0 1 0 0 1 1 1 0 1 1
gene no.1 = 0 0 1 1 0 0 0 0 1 1 1 1
gene no.2 = 1 0 1 0 1 1 1 0 0 1 0 1

～省略～

>> generation no.0
>> max gene[15] =   0  1  1  1  0  1  1  0  1  0  1  1
>> Fitness = 390
>> generation no.1
>> max gene[27] =   0  1  0  0  0  1  1  1  1  1  1  1
>> Fitness = 395
>> generation no.4
>> max gene[23] =   0  0  0  1  1  1  1  0  1  1  1  1
>> Fitness = 420
>> generation no.9
>> max gene[7] =   0  0  1  1  1  1  0  0  1  1  1  1
>> Fitness = 425
>> generation no.11
>> max gene[19] =   0  1  1  1  1  1  0  0  1  1  1  1
>> Fitness = 445
grep ">> max_fitness" stdout > stdout.max
tail stdout > stdout.end
```

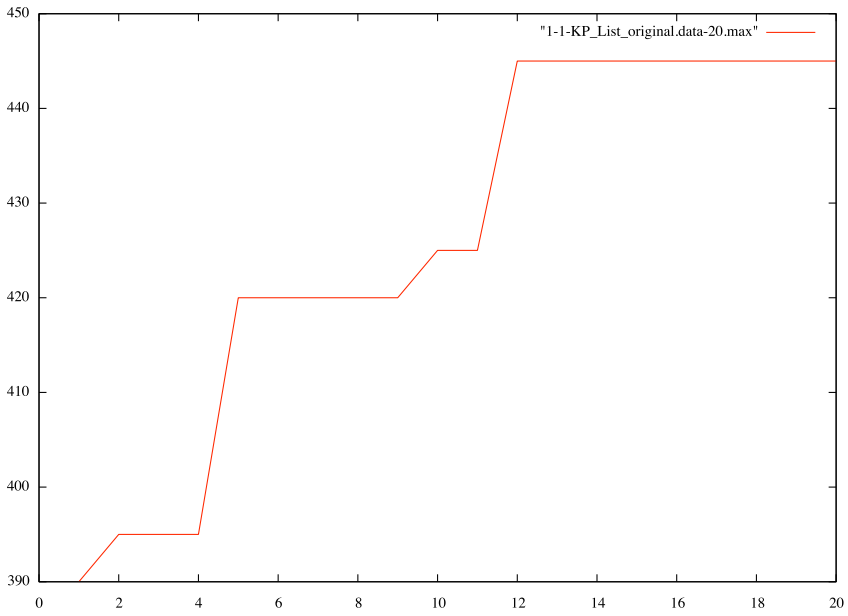

図 1: 全て実装した場合

全て実装した場合、適応度が上手く増加していることが分かる。また、この場合での最大の適応度が全パターンの中で最大となった。

2.2　選択、交叉を実装した場合

以下に選択、交叉を実装した場合の実行結果を示す。

実行結果

```
[j05002@ga-knapsack]% ./run_ga.sh 1 1 KP_List_original.data 20
# reading args => seed_pop=1, seed_ga=1, itemfile=KP_List_original.data,
max_generation=20
gene no.0 = 0 1 0 1 0 0 1 1 1 0 1 1
gene no.1 = 0 0 1 1 0 0 0 0 1 1 1 1
gene no.2 = 1 0 1 0 1 1 1 0 0 1 0 1

～省略～

>> generation no.0
>> max gene[15] =   0  1  1  1  0  1  1  0  1  0  1  1
>> Fitness = 390
>> generation no.1
>> max gene[27] =   0  1  0  0  0  1  1  1  1  1  1  1
>> Fitness = 395
>> generation no.3
>> max gene[12] =   0  0  1  0  0  1  1  1  1  1  1  1
>> Fitness = 405
>> generation no.4
>> max gene[36] =   0  1  1  0  1  1  0  1  1  0  1  1
>> Fitness = 420
>> generation no.10
>> max gene[19] =   0  0  1  0  1  1  0  1  1  1  1  1
>> Fitness = 430
grep ">> max_fitness" stdout > stdout.max
tail stdout > stdout.end
```

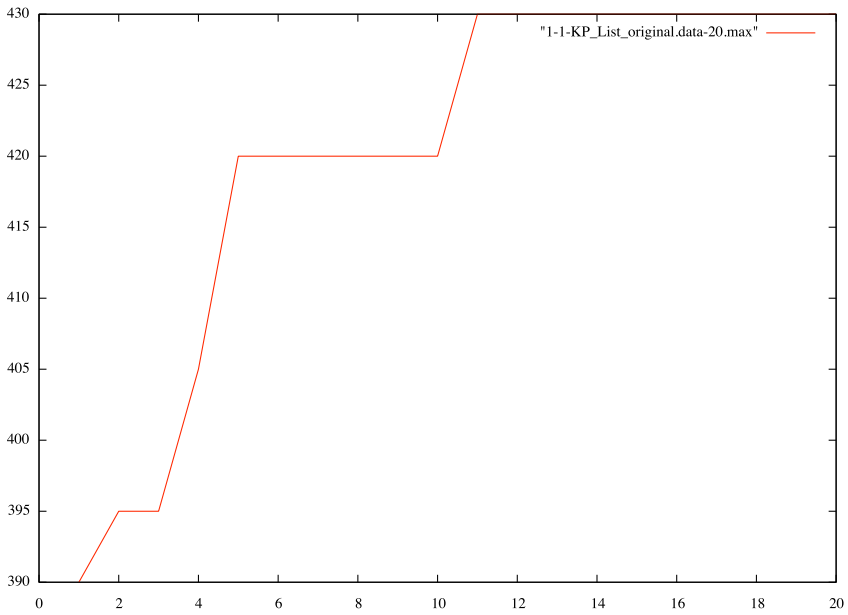

図 2: 選択、交叉を実装した場合

選択、交叉を実装した場合の結果を見てみると、全て実装したときのように上手く適応度が上がっている。この場合、突然変異が起こらないという条件だが、もともと突然変異が起こる確率は 10%なので、頻繁に起こるわけでは無い。したがって、全て実装した場合と似た形で適応度が上がっていると考えられる。

2.3 交叉、突然変異を実装した場合

以下に交叉、突然変異を実装した場合の実行結果を示す。

実行結果

```
[j05002@ga-knapsack]% ./run_ga.sh 1 1 KP_List_original.data 20
# reading args => seed_pop=1, seed_ga=1, itemfile=KP_List_original.data,
max_generation=20
gene no.0 = 0 1 0 1 0 0 1 1 1 0 1 1
gene no.1 = 0 0 1 1 0 0 0 0 1 1 1 1
gene no.2 = 1 0 1 0 1 1 1 0 0 1 0 1

～省略～

>> generation no.0
>> max gene[15] =   0  1  1  1  0  1  1  0  1  0  1  1
>> Fitness = 390
>> generation no.4
>> max gene[38] =   0  1  0  1  1  1  0  1  1  1  1  0
>> Fitness = 395
>> generation no.6
>> max gene[17] =   0  1  1  1  1  0  1  0  1  1  1  1
>> Fitness = 405
grep ">> max_fitness" stdout > stdout.max
tail stdout > stdout.end
```

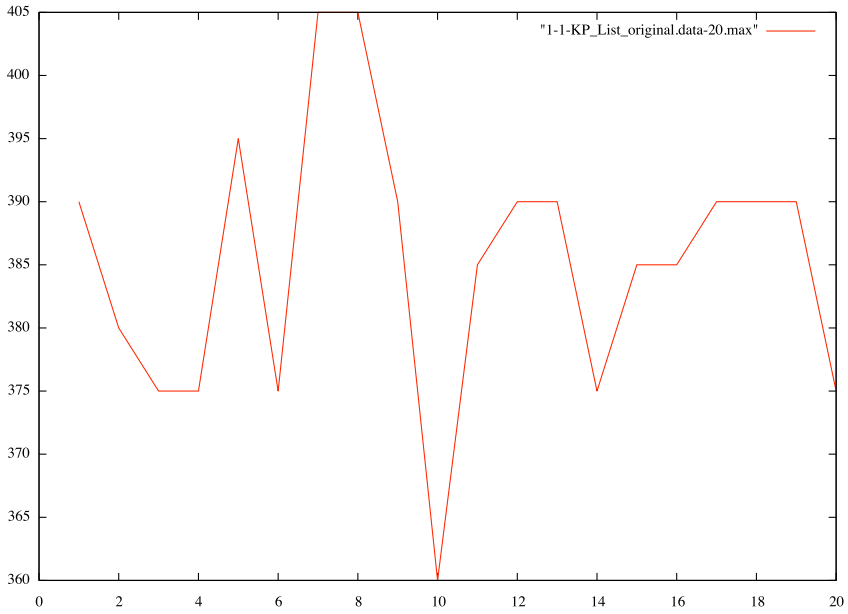

図 3: 交叉、突然変異を実装した場合

交叉、突然変異を実装した場合の結果を見ると、適応度の上下が激しくなっているのが分かる。この場合、選択が行われないので適応度の高い遺伝子が次の世代に残され、適応度の低い遺伝子が淘汰されるということが行われない。その状態で交叉、突然変異が起こるので、このような結果になったと考えられる。

2.4 選択、突然変異を実装した場合

以下に選択、突然変異を実装した場合の実行結果を示す。

実行結果

```
[j05002@ga-knapsack]% ./run_ga.sh 1 1 KP_List_original.data 20
# reading args => seed_pop=1, seed_ga=1, itemfile=KP_List_original.data,
max_generation=20
gene no.0 = 0 1 0 1 0 0 1 1 1 0 1 1
gene no.1 = 0 0 1 1 0 0 0 0 1 1 1 1
gene no.2 = 1 0 1 0 1 1 1 0 0 1 0 1

～省略～

>> generation no.0
>> max gene[15] =   0  1  1  1  0  1  1  0  1  0  1  1
>> Fitness = 390
>> generation no.5
>> max gene[19] =   0  1  1  0  1  1  1  0  1  0  1  1
>> Fitness = 400
>> generation no.9
>> max gene[6] =   0  1  0  0  1  1  0  1  1  1  1  1
>> Fitness = 420
grep ">> max_fitness" stdout > stdout.max
tail stdout > stdout.end
```

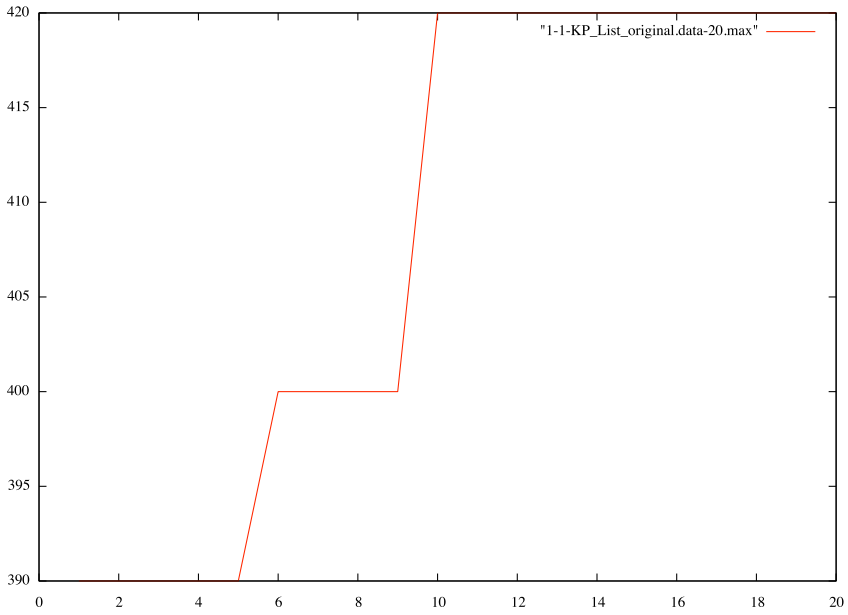

図 4: 選択、突然変異を実装した場合

選択、突然変異を実装した場合の結果を見てみると、なかなか上手く適応度が上昇しているように見える。しかし本来ならば交叉が行われないので、現在の世代において適応度の高い遺伝子が次世代にそのまま残るので適応度は、突然変異が起こらない限り一定になるはずである。今回の場合だと、第五世代と第六世代の間、および第九世代と第十世代の間で突然変異が起こり、たまたまうまい具合に突然変異し、適応度が高くなっただけだと考えられる。

2.5　選択のみ実装した場合

以下に選択のみ実装した場合の実行結果を示す。

ソースコード

```
[j05002@ga-knapsack]% ./run_ga.sh 1 1 KP_List_original.data 20
# reading args => seed_pop=1, seed_ga=1, itemfile=KP_List_original.data,
max_generation=20
gene no.0 = 0 1 0 1 0 0 1 1 1 0 1 1
gene no.1 = 0 0 1 1 0 0 0 0 1 1 1 1
gene no.2 = 1 0 1 0 1 1 1 0 0 1 0 1

～省略～

>> generation no.0
>> max gene[15] =   0  1  1  1  0  1  1  0  1  0  1  1
>> Fitness = 390
grep ">> max_fitness" stdout > stdout.max
tail stdout > stdout.end
```

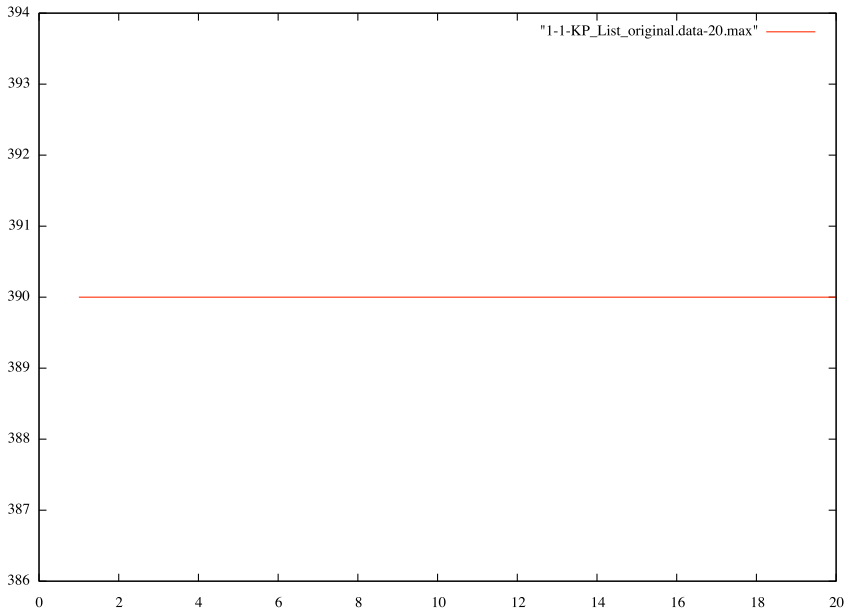

図 5: 選択のみ実装した場合

選択のみ実装した場合の結果を見ると、適応度が一定になっている。選択しか行われないということは、現在の世代において適応度の高い遺伝子が次世代に残り、さらに数が増えるだけなので適応度は変わらない。最終的に、集団の中には１種類の遺伝子だけしか存在しなくなるはずである。

2.6　交叉のみ実装した場合

以下に交叉のみ実装した場合の実行結果を示す。

ソースコード

```
[j05002@ga-knapsack]% ./run_ga.sh 1 1 KP_List_original.data 20
# reading args => seed_pop=1, seed_ga=1, itemfile=KP_List_original.data,
max_generation=20
gene no.0 = 0 1 0 1 0 0 1 1 1 0 1 1
gene no.1 = 0 0 1 1 0 0 0 0 1 1 1 1
gene no.2 = 1 0 1 0 1 1 1 0 0 1 0 1

～省略～

>> generation no.0
>> max gene[15] =   0  1  1  1  0  1  1  0  1  0  1  1
>> Fitness = 390
>> generation no.3
>> max gene[17] =   0  1  1  1  1  0  1  0  1  1  1  1
>> Fitness = 405
>> generation no.4
>> max gene[19] =   0  1  1  1  0  1  0  1  1  0  1  1
>> Fitness = 410
>> generation no.15
>> max gene[15] =   0  1  1  1  0  1  1  0  1  1  1  1
>> Fitness = 420
grep ">> max_fitness" stdout > stdout.max
tail stdout > stdout.end
```

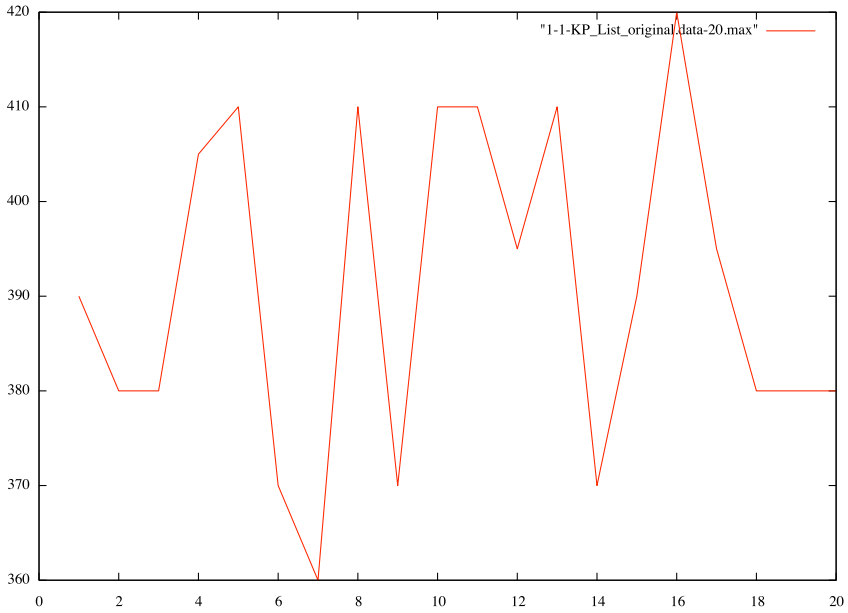

図 6: 交叉のみ実装した場合

交叉のみ実装した場合の結果を見てみると、適応度の上下が激しいことが分かる。初期集団の遺伝子が、どんどん交叉されて変化するだけなので、適

応度が安定することが無い。したがって、このような結果になったと考えられる。

2.7 突然変異のみ実装した場合

以下に突然変異のみ実装した場合の実行結果を示す。

ソースコード

```
[j05002@ga-knapsack]% ./run_ga.sh 1 1 KP_List_original.data 20
# reading args => seed_pop=1, seed_ga=1, itemfile=KP_List_original.data,
max_generation=20
gene no.0 = 0 1 0 1 0 0 1 1 1 0 1 1
gene no.1 = 0 0 1 1 0 0 0 0 1 1 1 1
gene no.2 = 1 0 1 0 1 1 1 0 0 1 0 1

～省略～

>> generation no.0
>> max gene[15] =   0  1  1  1  0  1  1  0  1  0  1  1
>> Fitness = 390
>> generation no.14
>> max gene[16] =   0  1  0  1  1  0  0  1  1  1  1  1
>> Fitness = 395
grep ">> max_fitness" stdout > stdout.max
tail stdout > stdout.end
```

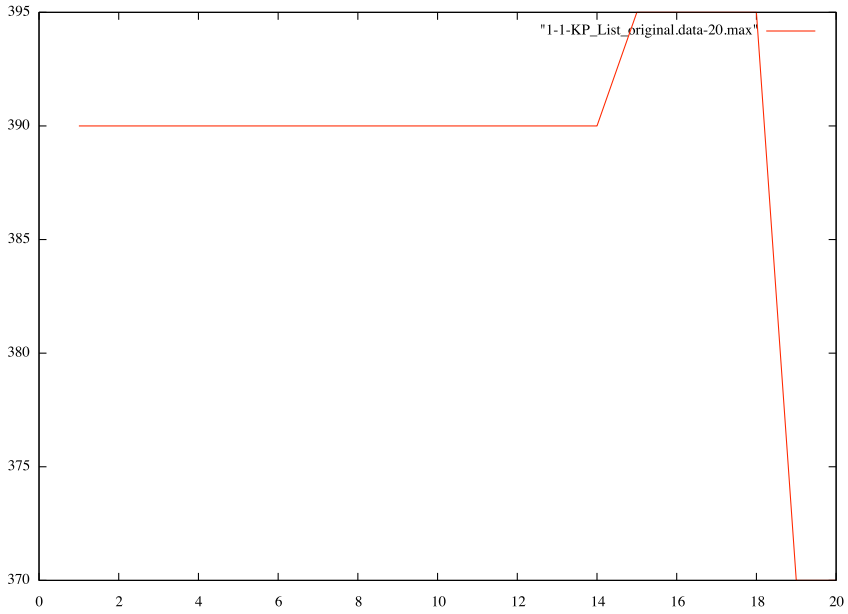

図 7: 突然変異のみを実装した場合

突然変異のみを実装した場合の結果を見ると、途中まで適応度が安定して、後半に変化していることが分かる。突然変異しか起こる可能性は無いので、

安定している所は突然変異が起こっていないということを示している。また、第十四世代と第十五世代の間、および第十八世代と第十九世代の間で適応度が変化しているので、その二カ所で突然変異が起きていることが分かる。突然変異が起こる可能性は 10%なので、終了世代数を大きな値にした場合、安定状態が長く続き、たまに (10%の確率で) 適応度が上下するという結果になるはずである。

そこで、終了世代数を 3000 にして実行してみた。
以下にその結果を示す。

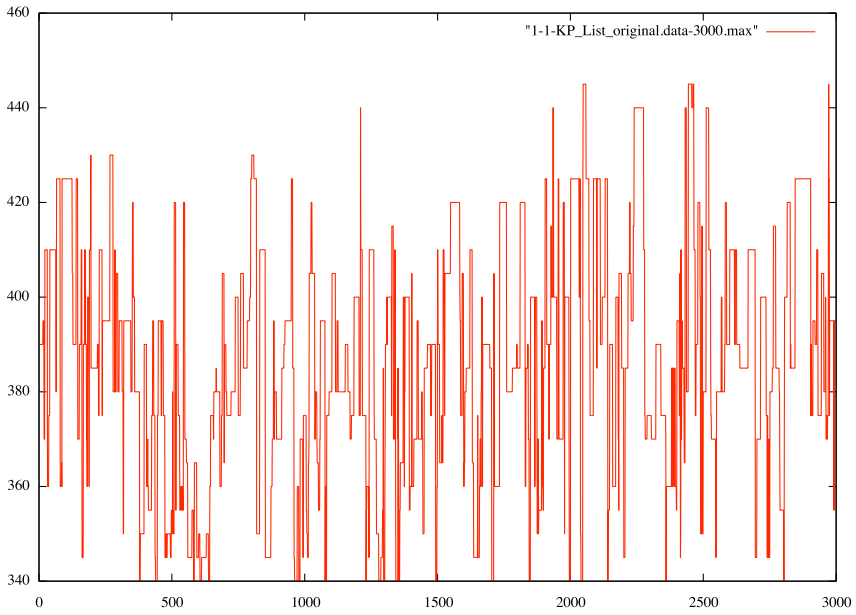

図 8: 終了世代数 3000

縦に激しく変化しているのであまり安定していないように見えるが、安定状態が長く続いているはずである。予想していたより判断しにくい結果となった。

3 全体の考察

これまでの結果を見ると、選択は適応度を高く保ち安定させようとする働きがあり、交叉は適応度を変化させようとする働きがあり、突然変異は適応度の安定し膠着した状態を変化させる働きがあると考えられる。